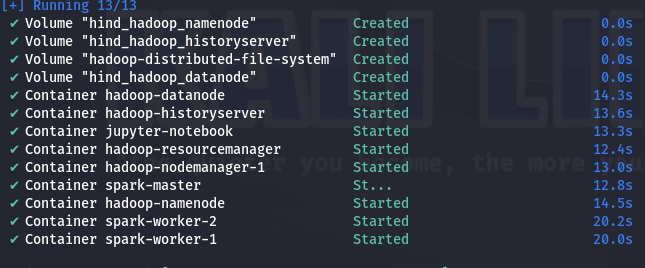
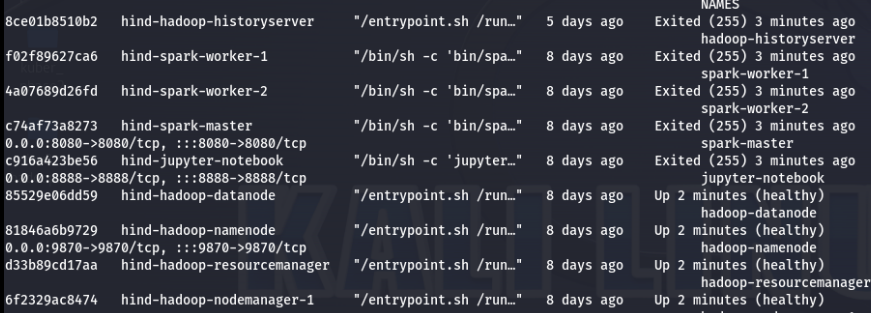
محمد حسین شهبازی گزارش تمرین سوم فاز 0 و 1 40031088

فاز صفر:

نمایش موفقیت آمیز ساخت کانتینر ها:  


نمایش کانتینر های ایجاد شده:



توضیح کانتینر های ایجاد شده:

NameNode: گره اصلی در HDFS Hadoop که ابرداده سیستم(metadata) فایل را نگهداری و مدیریت می کند.

DataNode: گره‌های ذخیره کننده داده در HDFS که بلوک‌هایی از داده‌ها را زمانی که NameNode به آنها گفته می‌شود ذخیره و بازیابی می‌کنند.

ResourceManager: مرجع مرکزی در YARN برای مدیریت منابع و زمان‌بندی کار ها.

NodeManager: مدیر فریمورک هر ماشین در YARN که مسئول کانتینرها، نظارت بر استفاده از منابع آنها و گزارش همان به ResourceManager است.

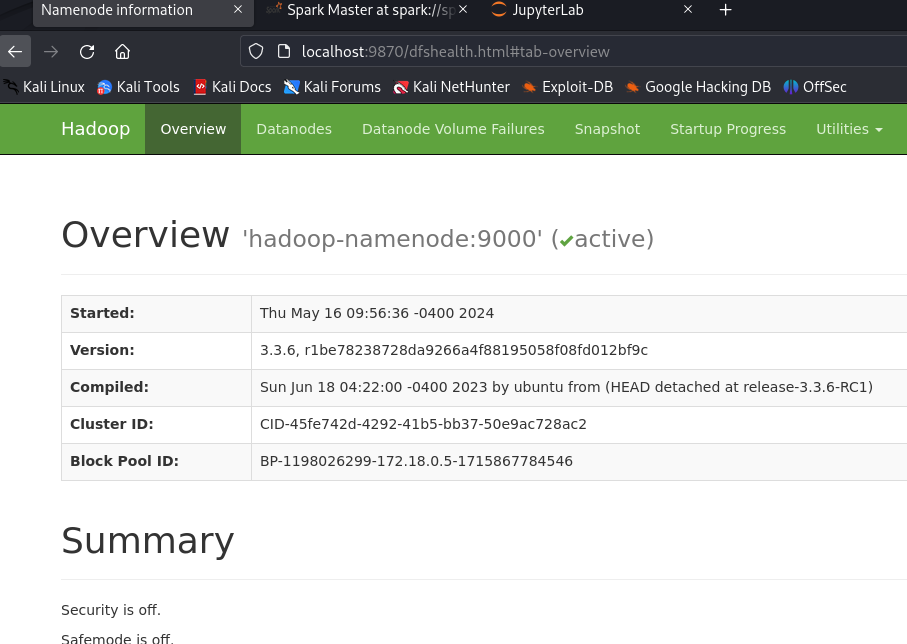
Spark Master: نقطه مرکزی و نقطه ورود خوشه Spark. مسئولیت توزیع وظایف به Sparkworker را بر عهده دارد.

Spark Worker: گره های پردازنده در کلاستر Spark که وظایف توزیع شده توسط Spark Master را اجرا می کنند.

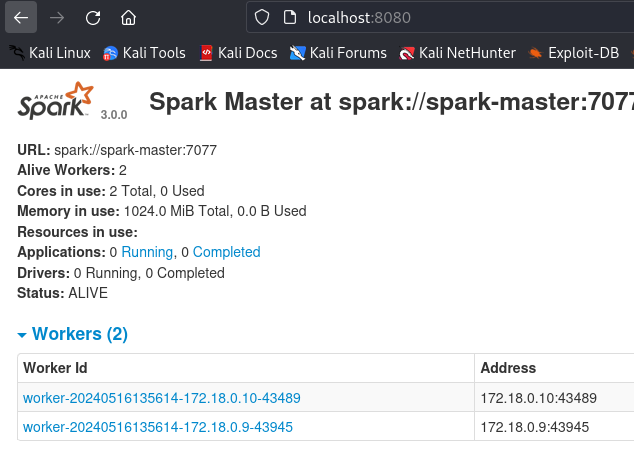
Jpyter Notebook: یک برنامه وب منبع باز که امکان ایجاد و به اشتراک گذاری اسناد حاوی live code، Visualization را فراهم می کند. اغلب با Spark برای تجزیه و تحلیل داده ها استفاده می شود.

نمایش UI برای Hadoop و Spark و Jupyter :

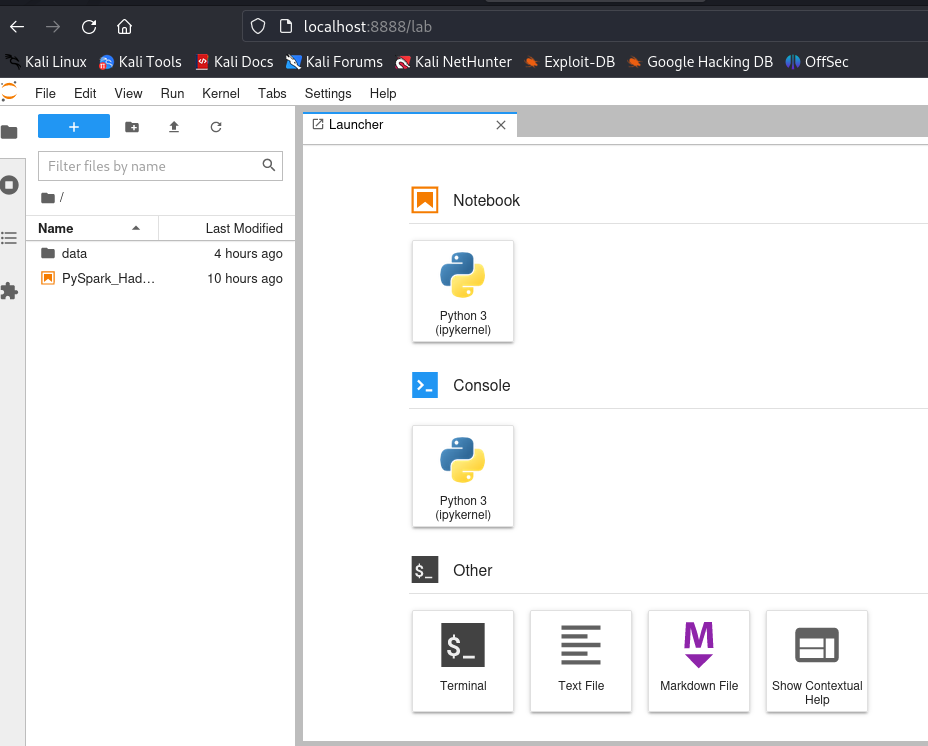
Hadoop:



:Spark



:Jupiter notebook



**توضیحات مربوط به تعداد نودهای اسپارک و منابع استفاده شده:**

تعداد نود های اسپارک: 3 عدد، یک master و 2 worker

منابع استفاده شده: 2 هسته پردازنده و 1024 مگابایت رم

**توضیحات مربوط به اطلاعات NameNode و فایل سیستم آن:**

128 فایل و دایرکتوری، 83 بلوک (83 بلوک تکراری، 0 گروه بلوک کدگذاری شده پاک شده) = 211 شیء سیستم فایل کل.

در دایرکتوری : /hadoop/dfs/name

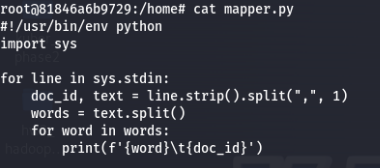
IMAGE\_AND\_EDITS Type =

StorageType: DISK

با ظرفیت 58.82 گیگابایت

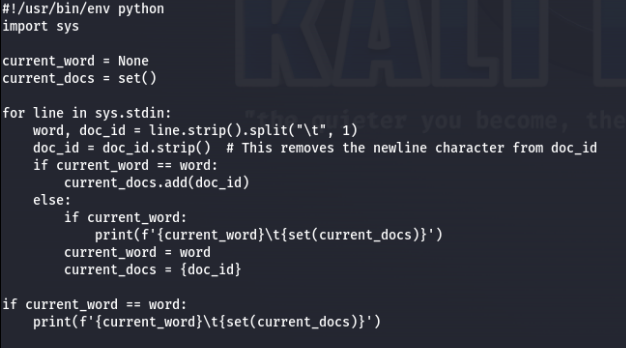
فاز 1:

بخش اول:

نمایش و توضیح کد mapper بخش اول:  


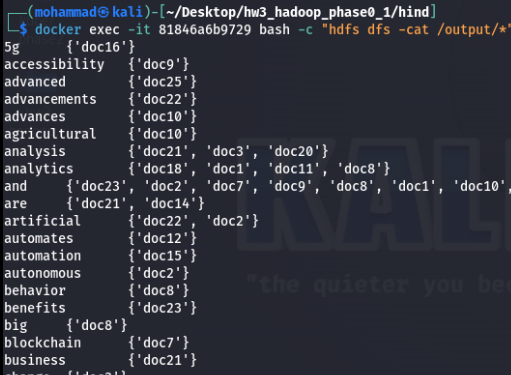
توضیح: ابتدا خود Hadoop فایل input.txt در sys.stdout میریزد. سپس در اینجا از فایل خوانده شده از sys.stdin خط به خط خوانده شده و بین doc id و جمله روبروی آن که با کاما از هم جدا شده اند با دستور split() جدا میشوند و اسپیس های اضافی حذف میشوند و در نهایت با دستور print() در sys.stdout ریخته میشوند.

نمایش و توضیح کد reducer.py :



توضیح:در اینجا فایل ایجاد شده در قسمت قبل خط به خط از sys.stdin خوانده شدن و اگر کلمه دوباره در یک doc id دیگر تکرار شده بود آن را به مجموعه doc id هایش اضافه میکند و اگر کلمه جدید بود آن را به مجموعه doc id های جدید اضافه میکند.

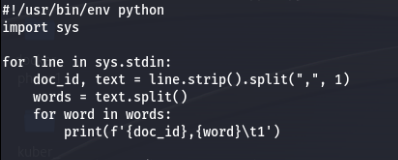
خروجی :



توضیح:همانطور که مشاهده میشود کلمات در کنار لیست ( (Setاز doc id های خودش قابل تفکیک و مشاهده است.

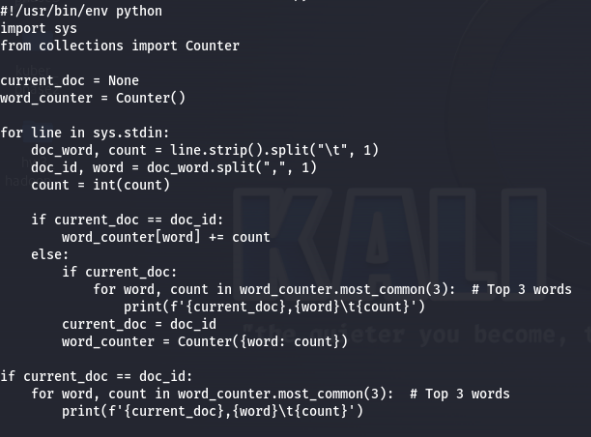
بخش دوم:

نمایش و توضیح کد mapper بخش دوم:



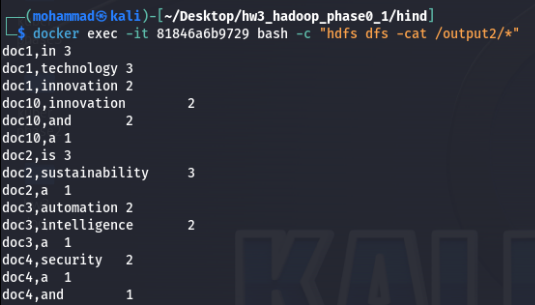
توضیح: ابتدا خود Hadoop فایل input.txt در sys.stdout میریزد. سپس در اینجا از فایل خوانده شده از sys.stdin خط به خط خوانده شده و بین doc id و جمله روبروی آن که با کاما از هم جدا شده اند با دستور split() جدا میشوند و اسپیس های اضافی حذف میشوند و در نهایت با دستور print() در sys.stdout ریخته میشوند.

نمایش و توضیح کد reducer.py :



توضیح:

ابتدا یک counter() برای شمارش 3= kبرترین کلمه پر تکرار ایجاد شده ایجاد میکنیم و خط به خط از sys.stdin ، doc id ها و کلمه روبرویش را میخوانیم و به تعداد تکرار آن اضافه میکنیم و سپس زمانی که برای یک doc id این کار تمام شد با دستور most\_common(3) که در کلاس counter وجود داشت 3 برترین کلمه را پیدا کرده و در خروجی چاپ میکنیم.

خروجی برای k=3:  


توضیح: همانطور که مشاهده میشود در خروجی doc id و شماره تکرار هر کلمه روبروی آن به تفکیک مشخص شده است.